DOCUMENT RESUME

ED 261 868 SE 045 995

TITLE Computer Science Curriculum Guide. Bulletin 1610. Revised.
INSTITUTION Louisiana State Dept. of Education, Baton Rouge. Div. of Academic Programs.
PUB DATE 83
NOTE 42p.; For other guides in this series, see SE 045 987.
PUB TYPE Guides - Classroom Use - Guides (For Teachers) (052)

EDRS PRICE MF01/PC02 Plus Postage.
DESCRIPTORS *Behavioral Objectives; Computer Literacy; *Computer Science; *Computer Science Education; *Course Content; Course Descriptions; Curriculum Development; *Learning Activities; Mathematics Curriculum; Mathematics Education; *Programing; Secondary Education; Secondary School Mathematics; State Curriculum Guides
IDENTIFIERS *Louisiana

ABSTRACT

This curriculum guide for computer science was developed to establish statewide curriculum standards for the Louisiana Competency-based Education Program. It consists of: (1) a rationale for and overview of the secondary school mathematics program; (2) a list of five suggestions for using the guide; (3) programming, debugging, and documentation standards; (4) a list of six course goals; and (5) a curriculum outline which contains performance objectives for five topic areas (computer literacy, algorithm construction and flowcharting, steps in writing a program, programming procedures, and optional advanced topics); (6) lists of activities arranged by major topic area; and (7) a list of additional suggested activities. A short bibliography, a list of supplementary materials, and brief comments on evaluative techniques in computer science are also provided. (JN)

****
* Reproductions supplied by EDRS are the best that can be made *
* from the original document. *
****

SE

U.S. DEPARTMENT OF EDUCATION
NATIONAL INSTITUTE OF EDUCATION
EDUCATIONAL RESOURCES INFORMATION CENTER (ERIC)
✓ This document has been reproduced as received from the person or organization originating it.
☐ Minor changes have been made to improve reproduction quality.
• Points of view or opinions stated in this document do not necessarily represent official NIE position or policy.

ED261868

STATE OF LOUISIANA
DEPARTMENT OF EDUCATION

COMPUTER SCIENCE CURRICULUM GUIDE

BULLETIN 1610

Revised 1983

SE 045995

THOMAS G. CLAUSEN, Ph.D.
Superintendent

"PERMISSION TO REPRODUCE THIS MATERIAL HAS BEEN GRANTED BY
S. Ebarb
TO THE EDUCATIONAL RESOURCES INFORMATION CENTER (ERIC)."

This public document was published at a cost of $1.37 per copy by the Division of Administration, Administrative Services, P.O. Box 44095, Baton Rouge, Louisiana 70804 to fulfill the requirements of La. R.S. 17:24 (E) to develop and establish state wide curriculum standards for required subjects. This material was printed in accordance with the standards for printing by state agencies established pursuant to R.S. 43:31.

STATE OF LOUISIANA

DEPARTMENT OF EDUCATION

COMPUTER SCIENCE CURRICULUM GUIDE

BULLETIN 1610

Revised

1983

Issued by

OFFICE OF ACADEMIC PROGRAMS

THOMAS G. CLAUSEN, Ph.D.

Superintendent

FOREWORD

Curriculum guides have been developed for grades K-8 at the elementary level and for each mathematics course at the secondary level. These guides represent the best thinking of a selected statewide committee established to determine the scope of mathematics content which should be taught at each level.

The mathematics curriculum guides are another segment of the total educational program established by this administration and mandated by the Legislature in both the accountability and assessment and the competency-based education laws. This educational program requires that specific skills and concepts be established for each grade level and for each subject area. The mathematics curriculum guides with course outlines, performance objectives, and coordinated activities effect this phase of the program.

It is hoped that the mathematics curriculum guides will make a major contribution to the improvement of mathematics instruction in the schools of Louisiana.

Thomas G. Clausen, Ph.D.

LOUISIANA STATE BOARD

OF ELEMENTARY AND SECONDARY EDUCATION

Bro. Felician Fourrier, S.C.
President
Member-at-Large

Dr. Claire R. Landry
Vice-President
First Congressional District

Mrs. Marie Louise Snellings
Secretary-Treasurer
Fifth Congressional District

Mr. Jesse H. Bankston
Sixth Congressional District

Mr. Milton Hamel
Fourth Congressional District

Ms. Gloria Harrison
Member-at-Large

Ms. Martha Scott Henry
Member-at-Large

Dr. John A. Bertrand
Seventh Congressional District

Mr. A. J. "Sookie" Roy, Jr.
Eighth Congressional District

Bro. David Sinitiere, F.S.C.
Second Congressional District

Mr. Jack Pellegrin
Third Congressional District

EXECUTIVE DIRECTOR

James V. Soileau
State Board of Elementary
and Secondary Education

ACKNOWLEDGMENTS

The Statewide Mathematics Curriculum Committee is to be commended for its work in the development of the Mathematics Curriculum Guide Series, K-12. Leadership for this project was provided by Dr. Jean Reddy Clement, Section Chief, Mathematics Section, Bureau of Secondary Education.

Supervisors in the Bureau of Elementary Education working under the direction of Mrs. Bonnie Ross, Elementary Supervisor, developed the activities for the K-8 guide. The activities for the secondary mathematics guides were written by a committee of secondary mathematics teachers and Dr. Clement. Revisions were under the direction of Dr. Clement and Mrs. Doris Meyer. These dedicated educators are to be commended for their enthusiasm in undertaking this formidable project and for the superb quality of their contributions to this unique and comprehensive Mathematics Curriculum Series.

William E. Stephens, Jr.
Assistant Superintendent
Office of Academic Programs

Helen Brown, Ed.D.
Director
Bureau of Curriculum, Inservice, and Staff Development

Joe C. Rivet
Director
Bureau of Secondary Education

STATEWIDE MATHEMATICS CURRICULUM COMMITTEE

Dr. Jean Reddy Clement, Chairman
Section Chief/Mathematics
State Department of Education
Post Office Box 44064
Baton Rouge, Louisiana 70804

Dr. Jane Abshire
Mathematics Supervisor
Vermilion Parish School Board
Abbeville, Louisiana 70510

Mrs. Ruth Atherton
Baton Rouge Magnet School
Baton Rouge, Louisiana 70806

Mrs. Annette Ballard
Elementary Consultant
Calcasieu Parish School Board
Lake Charles, Louisiana 70815

Dr. Myrna L. Bond
1320 Brocade Street
Baton Rouge, Louisiana 70815

Mrs. Olympia Boucree
Mathematics Supervisor
Orleans Parish School Board
New Orleans, Louisiana 70122

Mrs. Patsy Ann Bullock
Glen View Junior High School
Ruston, Louisiana 71270

Mr. James E. Ferguson
Ruston High School
Ruston, Louisiana 71270

Mrs. June Harper
McKinley Middle School
Baton Rouge, Louisiana 70802

Mrs. Suanne Jacobs
Sam Houston High School
Lake Charles, Louisiana 70601

Mrs. Jane Johnston
West Monroe High School
West Monroe, Louisiana 71291

Mrs. Margaret Kennedy
Grand Lake Elementary School
Lake Charles, Louisiana 70601

Mrs. Ida V. King
West Monroe High School
West Monroe, Louisiana 71291

Mrs. Marian King
Istrouma High School
Baton Rouge, Louisiana 70805

Mrs. Pearl Leach
Cameron Parish School Board
Cameron, Louisiana 70631

Mr. Lewis C. Martin
Epps High School
Epps, Louisiana 71237

Ms. Theresa M. Martinez
South Cameron High School
Creole, Louisiana 70632

Mr. Otto Sellers
Captain Shreve High School
Shreveport, Louisiana 71105

Mrs. Patricia Valentine
Kiroli Elementary School
West Monroe, Louisiana 71291

Mr. Henry Wilson
Transylvania Elementary School
Lake Providence, Louisiana 71286

Dr. Elton Womack
Post Office Box 97
Hall Summit, Louisiana 71034

ACTIVITIES COMMITTEE

Secondary Mathematics Curriculum Guide

Dr. Jean Reddy Clement, Chairman
Section Chief/Mathematics
Louisiana State Department
of Education
Post Office Box 44064
Baton Rouge, Louisiana 70804

Mrs. Ruth Atherton
Baton Rouge Magnet School
Baton Rouge, Louisiana 70806

Mrs. Sylvia Dunn
Nutrition Supervisor
Louisiana State Department
of Education
Post Office Box 44064
Baton Rouge, Louisiana 70804

Dr. Al Fabre
L.S.U. Laboratory School
Baton Rouge, Louisiana 70803

Dr. Jack Garon
L.S.U. Laboratory School
Baton Rouge, Louisiana 70803

Mrs. Pearl Leach
Cameron Parish School Board
Cameron, Louisiana

Dr. Elton Womack
Post Office Box 97
Hall Summit, Louisiana 71304

PILOT COMMITTEE

Computer Science Curriculum Guide

Mrs. Eleanor Thomas, Chairman
Baton Rouge High School
Baton Rouge, Louisiana 70806

Mr. Jim Burgard
East Jefferson High School
Metairie, Louisiana 70001

Mrs. Rosemary Denton
Carencro High School
Lafayette, Louisiana 70507

Ms. Gloria Love
Booker T. Washington High School
New Orleans, Louisiana 70125

Mr. Frank Moore
Archbishop Rummel High School
Metairie, Louisiana 70004

Mr. Walter Rush
Carroll High School
Monroe, Louisiana 71201

REVISION COMMITTEE

Dr. Jean Reddy Clement, Chairman
Louisiana State Department
of Education
Baton Rouge, Louisiana 70804

Ms. Sallie Deweese
Belaire High School
Baton Rouge, Louisiana 70815

Mrs. Mary Beth Johnson
West Monroe High School
West Monroe, Louisiana 71291

Mrs. Joycelyn Lee
Tioga High School
Tioga, Louisiana 71477

Mrs. Doris Meyer
Louisiana State Department
of Education
Baton Rouge, Louisiana 70804

Mr. Ted Mims
Louisiana State University
Baton Rouge, Louisiana 70803

Mr. Frank Moore
Archbishop Rummel High School
Metairie, Louisiana 70004
(504) 834-5592

Mrs. Claire Olivier
Andrew Jackson High School
Chalmette, Louisiana 70043

Ms. Kathy Ross
Jefferson Parish Central Office
Grenta, Louisiana 70053

Mrs. Joyce Sams
Ringgold High School
Ringgold, Louisiana 71608

Ms. Yvonne Tomlinson
Riverdale High School
Jefferson, Louisiana 70121

Mrs. Doris Voitier
Chalmette High School
Chalmette, Louisiana 70043

Ms. Sharon Whitehill
Belle Chasse High School
Belle Chasse, Louisiana 70037

LOUISIANA DEPARTMENT OF EDUCATION PERSONNEL

Bureau of Curriculum, Inservice, and Staff Development

Dr. Helen Brown, Director
Dr. Sylvia Torbet, Assistant Director
Mrs. Marteal Webb, Supervisor
Mr. Paul Vanderburg, Section Chief

Bureau of Secondary Education

Mr. Joe C. Rivet, Director
Dr. Jean Reddy Clement, Section Chief
Dr. David Gullatt, Supervisor
Mrs. Doris Meyer, Supervisor

INTRODUCTION

Act 750 of the 1979 Louisiana Legislature (R.S. 17:24.4) established the Louisiana Competency-Based Education Program. One of the most important provisions of Act 750 is the mandated "development and establishment of statewide curriculum standards for required subjects for the public elementary and secondary schools of this state . . ." The "statewide curriculum standards for required subjects" is defined as "the required subjects to be taught, curriculum guides which contain minimum skills and competencies, suggested activities, suggested materials of instruction, and minimum required time allotments for instruction in all subjects." Act 750 further provides that the "effective implementation date of the statewide curriculum standards for required subjects shall be the 1981-82 school year. Development of such curriculum shall begin by the 1979-80 school year."

During the 1978-79 school year, curriculum guides were developed by advisory and writing committees representing all levels of professional education and all geographic areas across the State of Louisiana for the following mathematics courses: Algebra I, Algebra II, Geometry, Advanced Mathematics, and Trigonometry. The major thrust of the curriculum development process in each of the guides has been the establishment of minimum standards for student achievement. Learning expectancies for mastery have been determined for each course/and or grade level. In addition, content outlines, suggested activities, procedures, and bibliographies have been developed as aids in support of the learning expectancies. The curriculum guides also contain activities designed to stimulate learning for those students capable of progressing beyond the minimums.

During the 1979-80 school year, the curriculum guides were piloted by teachers in school systems representing the different geographic areas of the State as well as urban, suburban, inner-city, and rural schools. The standard populations involved in the piloting reflected also the ethnic composition of of Louisiana's student population. Participants involved in the piloting studies used the curriculum guides to determine the effectiveness of the materials that were developed. Based upon the participants' recommendations at the close of the 1979-80 pilot study, curriculum guides were revised to ensure that they are usable, appropriate, accurate, comprehensive, relevant, and clear. These curriculum guides were implemented statewide in the 1980-81 school year. Following the established curriculum development procedures, curriculum guides for Mathematics I, Mathematics II, Consumer Mathematics, Business Arithmetic, and Computer Science were developed in 1979-80 and piloted in 1980-81. These curriculum guides were implemented statewide in 1981-82. This revision of the original guide has been prepared from suggestions collected statewide from teachers who have used the guide.

As curriculum guides are implemented, the following guidelines should prove helpful:

. . . curriculum standards should be considered as the foundation for the year's instructional program. Where other programs are already in operation, these curricular materials must be checked with the foundation curricula to ensure that appropriate course and/or grade level standards are included and maintained.

. . . curricular activities contained in the guides provide a number of suggestions for helping students to achieve the established standards. Activities to meet the needs of "average," "below average," and "above average" students have been included in the appropriate guides. These activities should prove helpful as the teacher plans and organizes instruction. Additional activities, however, may supplement or be used in lieu of those listed in the guide as long as these activities are designed to achieve similar specific objectives.

. . . curricular suggestions for meeting the needs of the special child have been prepared by the Division of Special Education. These suggestions are designed to provide help for teachers who work with special children in the regular classroom.

The continued effort of mathematics teachers to provide quality instruction will enhance our statewide goal to ensure that every student in the public elementary and secondary schools of the State of Louisiana has an opportunity to attain and to maintain skills that are considered essential to functioning effectively in society.

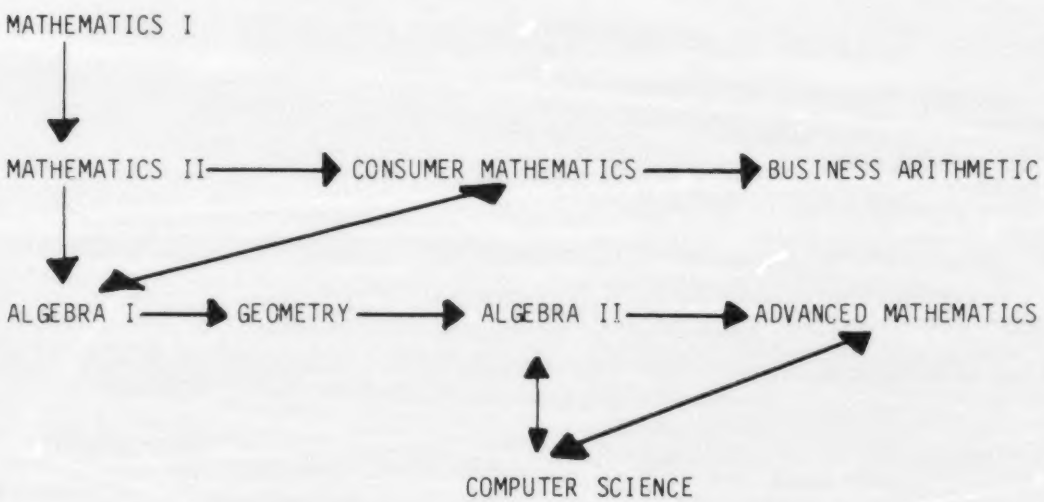
RATIONALE

Understanding the development of the entire set of mathematics curriculum guides is important to the proper use of the guides. This understanding is especially vital to the proper placement of students in the areas of Mathematics I, Mathematics II, Consumer Mathematics and Business Arithmetic. To avoid unnecessary duplication and repetition of content, the writing committee selected those topics which were deemed most appropriate for each of these courses. These topics were then eliminated from the content of the other courses or were treated with less emphasis.

Teachers and counselors need also to be aware of the difficulty levels of these courses. Mathematics I is the most fundamental course and is designed for those students entering ninth grade who have not acquired the basic skills in arithmetic. The stronger students who are still not quite prepared for success with Algebra I upon entering ninth grade should be encouraged to schedule Mathematics II. Mathematics II is designed to strengthen mathematical background and to prepare students for Algebra I and Geometry. Every student who plans to go to college should take Algebra I (at least). It is recommended that the college bound student also take Geometry and Algebra II.

Consumer Mathematics, as the name indicates, treats that mathematics which each of us encounters routinely as a citizen and consumer. The content differs from that of Business Arithmetic in that Business Arithmetic approaches the topics from the viewpoint of an employer or one engaged in business or manufacturing. It is not recommended that a student who has successfully completed Algebra II be allowed to take Mathematics I or Mathematics II.

The accompanying diagram should aid in understanding some possible avenues a student may take in his secondary mathematics career.



COMPUTER SCIENCE CURRICULUM GUIDE

DESIGN AND USE OF THIS GUIDE

1. Emphasis is placed on early hands-on experience as a motivational technique.

2. This guide is written in such a manner that any computer language could be employed.

3. No pacing chart is given, but the material is designed for a two semester course. The construction of programs is of primary importance.

4. The guide is written to be used sequentially.

5. Suggested content for a one semester course would be Sections I through IV J 3 with minimal coverage of Section I.

PROGRAMMING STANDARDS FOR COMPUTER SCIENCE STUDENTS

GENERAL RULES

The standards of Elements of Programming Style, by Kernighan and Plauger, should be used as a guide.

In addition:

1. Identifiers (names) must be chosen to reflect their function in the program.

2. Names should not be abbreviated excessively unless the language has limitations on name length.

3. Programs must be structured properly. Use subprograms whenever feasible.

4. The source program should be indented and formatted for maximum readability by a person.

5. GOTO statements (especially backward GOTO) should be avoided as far as possible. Any use of GOTO must be justifiable.

6. Use simple and direct constructions. Avoid "cute" tricks and kluges.

7. Write the program first in an easy-to-understand pseudo-language, then translate it into whichever language you have to use.

8. Don't patch bad code--rewrite it.

9. Make it right before you make it faster.

10. The program should check for bad input, such as inputting a string when a number is expected.

11. Output must always be properly labeled and in an easily readable format.

12. Do not start writing code immediately after you get the assignment. Think about the problem carefully first. Use the Top-Down approach and the full facilities of the host language.

DEBUGGING STANDARDS AND SUGGESTIONS

At least in the debugging stages, your program should produce immediate output to indicate the progress of the program. Some suggestions along this line appear later in this section. Never seek help with a program whose only output is an error message. Read the error message and make some attempt to understand the message.

Often students do not know the proper approach to use in debugging a syntactically correct program. At the heart of this is the fact that most students will sit down to ponder over the code and ask the question, "Why did my program die?"

The answer to this question is central to the methodology of debugging. Your program died because something happened in execution that caused it to die.

If this answer seems to be flippant or circular, think again. The original question has the same flavor as Aristotle's question about why objects fall to the ground. The question "Why?" is not the right question. The proper question to ask is "WHAT HAPPENED" in execution that caused it to die?"

Having asked the right question, debugging is straightforward. It is necessary to find out EXACTLY WHAT HAPPENED during the execution of the program. The wise programmer, when his program dies, takes a moment to consider the probable or possible causes, and then inserts the proper PUT/WRITE/PRINT/XPRNT/XDUMP statements that will give him a trace of the program's execution. Most errors can be found by such a trace, unless the programmer has a genuine misunderstanding of the language being used.

Many of the errors in programs involve either infinite loops or complicated branching. Very often the programmer thinks that he has coded several different branches for several different possibilities, but because of errors in the condition being tested, some of the branches are not taken. To find the errors, get a trace of the values of the variables relevant to the condition and the location in the program from which the output is being produced. In testing, it is especially important to remember that error-handling segments of code will actually be executed when the errors occur.

Another common source of errors, especially in modern structured programming, is that subprograms do not operate properly. The programmer, however, will waste time trying to debug (correct) a main program that is being fed spurious information from the subprogram. Get output showing the values passed as parameters to the subprograms, the values that actually get into the subprograms (in case there is some unwanted conversation because of missing declaration statements), the values of the parameters before they are passed back to the calling program, and the values finally received by the calling program.

In general, if debugging is difficult it is so because of a lack of information on the program. Assemble ALL the facts and the task should be relatively simple.

DOCUMENTATION STANDARDS

The following is an excerpt from the chapter entitled, "What are the Qualities of a Good Program?" from E. Yourdon's book Techniques of Program Structure and Design:

> "In my opinion, there is nothing in the programming field more despicable than an uncommented program. A programmer can be forgiven many sins and flights of fancy, including many of those listed in the sections below; however, no programmer, no matter how wise, no matter how experienced, no matter how hard-pressed for time, no matter how well-intentioned, should be forgiven an uncommented and undocumented program.
>
> If this seems an unreasonably venomous attack, you are invited to debug, maintain or change someone else's uncommented program. . .
>
> Nonexistent comments are an obvious problem; a more subtle problem can exist if the program is heavily laden with comments, but:
>
> 1. The comments are redundant.
> 2. The comments are obsolete.
> 3. The comments are (and always were) incorrect.
> 4. The comments are vague and imprecise.
> 5. The comments are correct, but incomplete.
> 6. The comments cannot be understood by anyone else.
>
> . . . Indeed, it is not surprising that some experiments have shown that it is faster and easier to fix bugs in someone else's program by first removing all of the comments."

The following are reasonable documentation standards:

1. A brief paragraph should be included at the beginning of the program describing what the program is about and the roles of the variables that you are using.

2. Comments should be included at key locations of the program to explain it.

3. Comments must be correct, including spelling and grammar.

4. Except for drawings that cannot be produced easily on a line printer, you should not write on your listing. Whatever you want to say should be incorporated into your program or documentation.

5. If the program has input data, the card format required for the data should be described in the documentation.

GOALS

During the course of Computer Programming, the student will:

1. Gain a basic understanding of the history and development of computers.

2. Learn basic design and functional units of a computer with emphasis on student operation of the computer.

3. Understand construction of algorithms and flowcharting techniques.

4. Know and apply the sequence of steps required in the writing of a structured computer program.

5. Understand the construction and uses of symbolic language in the creation of a structured program.

6. Understand the technological aspects of computer design and operation.

CURRICULUM OUTLINE AND PERFORMANCE OBJECTIVES

Recommended Course Content	Performance Objectives
I. Computer Literacy	
A. Methods of Calculations	A. To demonstrate a knowledge of the history of calculations, the student will be able to describe early methods of calculation, such as finger counting, abacus, and Chisanbop.
B. Manual Aids	B. To exhibit a knowledge of manual aids, the student will be able to explain Napier's Bones and the benefits of the credit/debit concept.
C. Mechanical Aids	C. To show a knowledge of early mechanical aids, the student will be able to list early mechanical aids, such as Pascal's Wheel Calculator and the early "Four Function" machines.
D. Punched Cards	D. To demonstrate an awareness of the significance of the punched card, the student will be able to: 1. Explain the advances that occurred with the advent of the punched cards and punched card machines. 2. State the contributions of Herman Hollerith.
E. Automatic Mechanical Aids	E. To exhibit a knowledge of automatic mechanical aids, the student will be able to: 1. Recognize the contributions of Babbage and others in the development of early mechanical calculators. 2. Identify types and methods of early analog calculating devices such as the slide rule.

Recommended Course Content	Performance Objectives
F. Computer Age	F. To demonstrate a knowledge of the advent of the computer age, the student will be able to: 1. Trace the generations of computer development: vacuum tube devices, transistors, integrated circuits and microprocessors. 2. Cite the contributions of John von Neumann.
G. Types of Computers	G. To illustrate a knowledge of the types of computers, the student will be able to: 1. Recognize and distinguish among three types of computers: digital, analog and hybrid. 2. State the benefits and limitations of digital and analog devices.
H. Hardware and Software	H. To demonstrate a knowledge of a computer system, the student will be able to define the terms hardware and software and be able to apply this knowledge to his own system.
I. Functional Units of a Computer	I. To demonstrate an understanding of the functional computer units, the student will be able to identify the types and operations of the following essential devices: 1. Input/output devices and their media (punched card, magnetic tape, typewriter, paper tape, disk) 2. Memories and types of memory devices (Ferrite Core, magnetic disk, etc.) and associated terminology (bit, byte, binary cell, address, etc.) 3. Control unit.

Recommended Course Content	Performance Objectives
J. Uses of Computers	J. To demonstrate a knowledge of the vast use of computers in society, the student will be able to state the basic uses of computers in business, information storage, scientific research and design, word processing, and the home.
II. Algorithm Construction and Flowcharting	
A. Problem Analysis	A. To illustrate a basic understanding of the procedure of problem analysis, the student will be able to use the proper steps in problem analysis: 1. Clearly define the problem to be solved or the task to be performed. 2. Reduce the solution of a collection of discrete steps. 3. Arrange the solution steps in a logical sequence.
B. Flowchart Symbols	B. To show a knowledge of symbols used in flowcharting, the student will be able to identify and use common flowchart symbols for the following: 1. Input/Output 2. Operations (Assignment) 3. Decision or Test 4. Direction of Flow 5. Initiation or Termination of a Process.
III. Steps in Writing a Program	To demonstrate the ability to write a program, the student will be able to:
A. Statement of Problem	A. Write a detailed statement of the problem to be solved or the task to be performed.
B. Algorithm and Flowchart	B. Construct a solution algorithm and flowchart.

Recommended Course Content	Performance Objectives
C. Write Program	C. Write statements to execute algorithm.
D. Trace Program	D. Check the program by stepping through the instruction sequence with representative data values.
E. Run Program	E. Run the program with the data to be processed.
F. Debug Program	F. Find and correct any syntax or logic errors.
G. Documentation	G. Document and label the program.
IV. Programming Procedures	To demonstrate proper programming procedures the student will be able to:
A. Available Languages	A. Cite examples of high level languages in current use and their acronyms (Fortran, Cobol, Basic, Pascal).
B. System Commands	B. Demonstrate the proper use of commands when operating the system. 1. Execute a program. 2. Make a listing of line. statements. 3. Save a program on tape or disk. 4. Load a program into the computer. 5. Erase memory in preparation for another program. 6. Use editing system to correct syntax or logic errors.
C. Remarks or Comments	C. Document a program with remark statements. 1. Introductory remarks include student name, title and description of program, and dictionary of variables. 2. Remarks within the body of the program

Recommended Course Content	Performance Objectives
D. Output (Literals)	D. Handle simple output. 1. Describe and identify a literal. 2. Use a simple statement to print a literal.
E. Variables	E. Select proper variable names and types from the following classifications: 1. Numeric (Real and Integer) 2. Alphanumeric 3. Alphabetic and Logical (if available on system)
F. Input/Output	F. Demonstrate a basic understanding of Input/Output. 1. List methods of inputting data, such as assignment, input statements, need statements, data files, etc. 2. Employ formatting techniques by using code to perform: a. Tab function b. Line suppression c. Print column headings
G. Assignment Statement and Elementary Library Functions	G. The student will be able to: 1. Write and identify valid assignment statements. 2. Use the integer, square root, absolute value, random number generator, and string functions.
H. Rational and Logical	H. The student will be able to: 1. Identify and use relational operators (less than, greater than, equal to, etc.). 2. Identify and use logical operators (and, or, not).

Recommended Course Content	Performance Objectives
I. Control Structures	I. The student will be able to exhibit a knowledge of the following types of structures: 1. It-then-else 2. While-do or repeat-until 3. Multiple selection (ON-GOTO or CASE) 4. Counted repetition (FOR-NEXT or DO-CONTINUE) 5. Use the various types of control structures in applications requiring counting procedures, summation statements, and neglected loops.
J. Subscripted Variables	J. To demonstrate an understanding of subscripted variables, the student will be able to handle: 1. Dimension Arrays 2. Input/Output Arrays 3. Sorting routines 4. Storing concatenation * 5. Operations with arrays (Matrix commands)
K. Subprograms	K. To exhibit a knowledge of subroutines and user-defined functions in structured programs, the student will be able to explain: 1. Purpose and Construction 2. Related commands
L. Graphics	L. To produce a graphic output, the student will be able to: 1. Plot Points and draw lines. 2. Use Graphics Characters if available. 3. Use programmed cursor movement if available. 4. Employ Peek and Poke statements.

*Optional

Recommended Course Content	Performance Objectives
M. Data Files	M. The student will be able to indicate a working knowledge of data files by: 1. Creating a data file 2. Reading data from a file 3. Using sequential files versus random access files 4. Updating data in a file
N. Advanced Programming Projects	N. To demonstrate programming expertise the student should be able to design, code, debug, and document a program that illustrates many of the above concepts.
V. Optional Advanced Topics	
A. Computer Design and Operation	A. To exhibit a knowledge of computer design and operation, the student will be able to: 1. Recognize and use the elements of symbolic logic. 2. Compute in the binary, decimal and hexadecimal (base 16) number systems. 3. Describe the operation of logic switching circuits.
B. Arithmetic-Logic	B. To demonstrate a knowledge of the arithmetic-logic components, the student will be able to: 1. Describe the functioning of instructions registers and data registers. 2. Outline the functioning of the arithmetic-logic units in the execution of mathematical tasks and data manipulations.
C. Machine Language	C. To illustrate an understanding of machine language, the student will be able to describe the binary structure of machine language.

Recommended Course Content	Performance Objectives
D. Assemblers and Compilers	D. To exhibit a knowledge of assemblers and compilers, the student will be able to: 1. Determine the meanings of source program and object program. 2. Explain the function of the compilers.
E. Assembly Language	E. The student will be able to write an elementary program in assembly language.

ACTIVITIES

Course Content Reference

I. Computer Literaccy

A. The teacher will demonstrate an abacus.

B. The teacher will construct Napier's Bones and demonstrate its use.

C. The teacher will obtain pictures of early mechanical devices; whenever possible, the devices themselves should be available to students.

D. (a) The teacher will explain the use of purchased cards as an input/output device.

(b) The student will translate punched holes into numeric and alpha numeric characters.

E. (a) The students can be assigned research in the form of reports from encyclopedias and other reference books on Pascal, Napier, Charles Babbage, Emile Jacquard, Herman Hollerth and John Powers and their contributions.

(b) The teacher can present additional analog devices, such as thermometer, pressure gauges, or an automobile speedometer.

(c) Student should be able to construct a slide rule to perform multiplication and division.

F. (a) The students can trace the parallel pattern of the computer and the radio and TV from the early large bulky vacuum tube types to the smaller transistor models.

(b) The teacher will discuss the problems of over-heating common in early computers.

(c) The teacher will provide resource materials for students to investigate von Neumann, Eckert and Mauchly, ENIAC, EDVAC, EDSAC.

G. (a) The teacher can compare the speedometer and odometer on an automobile to analog and digital computers so that students can relate to them.

(b) The student can list the things measured by analog devices: pressures, temperature, viscosity, etc.

Course
Content
Reference

G. (Continued)

(c) The teacher can demonstrate with computer output bills, statements from banks, department stores, utility companies, etc., the diversification of computer uses in business.

(d) The teacher will describe the capability of the computer to develop models and handle large quantities of data for scientific applications.

H. (a) The student will list types of hardware such as core, semiconductor memory, and various devices such as card readers, tape drives, CRT's, printers, etc.

(b) The student can list types of software such as assemblers, compilers, and operating systems.

I. (a) The student will identify input/output devices and their media (card, magentic tape, typewriter, paper tape).

(b) The student will identify memories and types of memory devices (Ferrite Core, magnetic disk, etc.) and associated terminology (bit, byte, binary cell, address, etc.).

(c) The teacher will provide "hands on" experience by letting students type an existing program into the computer.

(d) The student can use print memory statement to examine amount of memory space used in storage of a program.

(e) The teacher can demonstrate peripheral devices.

(f) Students can diagram the relationship between functional units.

J. (a) The teacher can demonstrate with computer output bills, statements from banks, department stores, utility companies, etc., the diversification of computer uses in business.

(b) The teacher will describe the capability of the computer to develop models and handle large quantities of data for scientific applications.

(c) The teacher will discuss the storage and retrieval capabilities in terms of the amount of time needed to find a record in a large file stored on the computer versus in a manual filing system.

Course
Content
Reference

II. Algorithm Construction and Flowcharting

A. (a) The teacher can list the steps in solving a problem, leaving out key steps. The student can then insert the missing step in its proper place.

(b) The teacher can assign simple problems, such as calculating the total sale including tax on an item when the price of each item is given, calculate the area of a circle when the radius is given, or find the third angle of a triangle.

B. (a) The student will be able to draw a flowchart using the proper symbols for simple algorithms that involve input/output processing (operations) and logic.

(b) Given a completed flowchart, the student will track it to determine the output.

(c) The student will be able to flowchart a nonmathematical process (answering the phone, a day at school).

III. Steps in Writing a Program

A. Given a problem, the student will decide if enough information has been given for a solution to the problem.

B. Given a program and flowchart, have the student decide if the program accomplishes what the flowchart intends.

C. The student will write a program which will execute the solution algorithm and produce the described output (pay careful attention to the proper use of the programming language being employed and the planned arrangement of input/output items).

D. The student will check the program by stepping through the instruction sequence with representative data values.

E. The student will run the program with the data to be processed.

F. The student will be able to find all errors in a given program.

G. The student will document, label, and sign the program, including a summary of the purpose and design of the program for future runs.

Course
Content
Reference

IV. Programming Procedures

A. (a) The teacher can describe and give examples of popular programming languages and their uses.

FORTRAN - Science, Mathematics, Statistics
COBOL - Business
BASIC - Beginners in Academic Environment
PASCAL - Highly Structured Academic Language

(b) The teacher can show the students programs written in different languages.

B. (a) The teacher can have students use the computer in command mode as a calculator.

(b) Given a program on tape or disk, the student will load, list, and run it.

(c) Have the student alter the program, then save it on tape or disk.

(d) Given a program with errors, the student should find and correct them.

(e) The student should be able to interpret error messages and employ proper techniques to correct them.

C. The student should be able to add remarks to an undocumented program.

D. (a) Given three statements, the student will be able to identify the one that will print out a literal.

(b) The student will be able to write a direct command to print a literal.

E. (a) Given a list of variables, the student should be able to determine if they are valid or invalid.

(b) The student should be able to select the proper variable name for reals, integers, or literals.

(c) Given an algebraic expression, the student will be able to write it in a form acceptable to the computer.

Course
Content
Reference

F. (a) The student will list methods for inputting data.

(b) Given data, the student should write a program to output results in tabular form (including column headings).

(c) The student should use tab functions to draw a simple design on display screen.

(d) The student should write a program to output results both vertically and horizontally.

(e) The language chosen for classroom use will determine which I/O specification should be used. Certain FORTRAN versions require no format statements, but specifications should be covered. Distinctions among integer, real and alpha fields are necessary in FORTRAN; COBOL specifications for fields used in calculation are important if that is the language used.

(f) Data specifications for alphabetic, numeric, and alpha-numeric data should be clearly delineated. Carriage and Printer Control are included in FORTRAN's format statements, whereas the Procedure Division output statements of COBOL cover the carriage control.

G. (a) Given an algebraic statement, the student will be able to write it in correct computer form.

(b) The student should make a list of all library functions available on his system.

(c) The student should write a program that employs several library functions (square root, absolute val[illegible], integer, random generator, string, etc.).

(d) The student should write a small program that does the job of a particular library function.

H. (a) The student can effectively express logic using symbols >, <, =, combined with "and," "or" and "not."

(b) The student should draw examples of switching circuits to illustrate "and", "or," and "not."

(c) The teacher should demonstrate logic circuits if equipment is available.

Course
Content
Reference

I. (a) Given a program with many unconditional branches, the student should revise the program.

(b) Given a program, the student will distinguish between unconditional and conditional branching.

(c) The student will employ logical operators to produce branching in a program.

(d) Given a loop within a program, the student will be able to identify its components (initialization, incrementation, body, stopping mechanism).

(e) The student should write a program to count the number of items in a data statement.

(f) Given a list of numbers, the student should write a program using a loop to produce their sum.

(g) Write a loop to create a pause or delay in output.

(h) Given a program with improperly nested loops, the student will correct it.

(i) Write a program using nested loops to simulate the operation of a digital clock.

J. (a) Given an array, the student will write an appropriate dimension statement.

(b) The student will write a program to print an array on the display screen.

(c) The teacher can show students examples of one or two dimensional arrays.

(d) The student can write instruction code to perform matrix operations. (optional)

(e) The student will write a statement to output a value located at a specific position in an array.

(f) After an explanation of bubble sorting, the student will write a program to arrange a list of answers or names in order.

(g) The student will join two or more strings and store or output the result.

Course
Content
Reference

K. (a) The student should be able to explain when a subroutine should be used.

(b) The student should write a program that incorporates a delay subroutine.

(c) The student should demonstrate a knowledge of subroutine commands by using them in a program.

(d) The student will write a program incorporating a user-defined function that is called upon more than once in a program.

L. (a) The student will write a program to draw a simple geometric shape by plotting points.

(b) The student should be able to write a program that will graph a function by plotting points.

(c) The student should write a program to do simple animation using graphic characters.

M. (a) The student will write a program to store data on disk or tape.

(b) The student will write a program to read data stored on disk or tape.

(c) The student will differentiate between sequential and random access files.

(d) Given a list of file programming situations, the student will decide which type of file is better.

N. Some possible projects include the following:

(a) Maintaining statistics for a school athletic team

(b) Payroll (gross pay, deductions, tax, etc.) for a mythical company

(c) Updating inventory, including total value of items, and flagging when re-orders are necessary

(d) Computer games, such as tic-tac-toe, NIM, slot machine, and blackjack

(e) A teaching program to provide drill in basic arithmetic. Computer explains rules, generates problems, keeps score.

Course
Content
Reference

V. Optional Advanced Topics

A. (a) The student can effectively express logic using symbols <, >, =, combined with "and," "or" and "not."

(b) The student should demonstrate the ability to express decimal numbers in binary and hexadecimal.

(c) The teacher may use the repetitive division method, keeping track of remainders to convert base 10 numbers to hex and binary.

(d) The teacher may choose to divide by powers of the desired base, keeping track of place values.

(e) The student should draw examples of switching circuits to illustrate "and" and "or."

(f) The teacher should demonstrate logic circuits if equipment is available.

B. (a) The teacher will illustrate how processing occurs by following instructions in the instruction register and accessing data in the data register. The process can be compared with following directions to assemble a toy or piece of furniture.

(b) The teacher will illustrate how certain data may be destroyed after execution of an operation.

(c) Add A to B--after execution the sum is located in B destroying the original data. The student should indicate how B could be protected if needed for future calculation.

C. The teacher will show students a machine language op code ef; 9c is a hex representation of 1001 1100. Hex is difficult for people; binary is even more difficult; therefore, mnemonic op codes (sound symbols) were developed which permit Add, Sub, etc.

D. (a) The student should draw a diagram to show how the source and object program are related.

(b) The teacher will point out that a compiler is merely a program written to translate an application program.

Course
Content
Reference

E. (a) Students will discuss a simple assembly language program that has been written for their system.

(b) Students will write an assembly language program to perform a simple task, such as adding two numbers.

ADDITIONAL SUGGESTED ACTIVITIES

1. Ideally ready access to a computer is desirable. Use of a computer at some remote site will encourage more careful debugging and programming since turnaround time will be delayed.

2. The BASIC language is probably the most suitable in that it can be taught quickly and was designed specifically for beginners. However, other language compilers will probably be more readily available. FORTRAN and/or COBOL can be used with equal success. It might be advantageous to the better student to teach PASCAL as this is the language which is used for advanced placement testing.

3. Flowcharting templates, coding forms, printer spacing charts, and record layout forms should be made available to the students, with their uses clearly defined.

4. Exercises arranging data for input and output can be assigned. The teacher can evaluate the arrangement according to aesthetics, readability, logical arrangement, field size, and appropriate headings.

5. Classroom activities in addition and subtraction in hexadecimal and binary should be given emphasis. The placement of this topic within the course should be at the teacher's discretion. Many textbooks place it in an appendix. If it is used, it is best to relate it to the expression of data within primary memory or on magnetic tape.

6. The hex code relates to EBCDIC (Extended Binary Coded Decimal Interchange Code). This is an 8 bit code. Certain manufactures employ a 6 bit (Binary Coded Decimal Code). Still others use ASCII (American Standard Code for Information Interchange).

7. Students can be encouraged to investigate the various computer job categories. Programmers, system analysts, managers, data control technicians, data entry technicians, and the duties/responsibilities associated with the positions are some of the areas.

8. Students should be made aware of the impact of the computer on society. There are vast amounts of data which must be handled more accurately because of population growth. There are negative effects which it is only fair to cite. These include the psychological effects of automation resulting in lack of identity and decline in morale of individuals, job security threatened by automation, inflexibility in providing information, and loss of privacy as a result of data accessibility.

9. Probably as a result of the rapid development of computer science, the terminology can be most confusing. Terms like "code" or "system" can have many meanings, yet other concepts may use several different terms. For example, "table," "array," and "matrix" all reference the same concept. Care should be taken to point out these peculiarities to students to avoid confusion.

BIBLIOGRAPHY

Awad, Elias A. Introduction to Computers in Business. Englewood Cliffs, New Jersey: Prentice-Hall., Inc., 1977.

Awad, Elias A. Business Data Processing. Englewood Cliffs, New Jersey: Prentice-Hall, Inc., 1971.

Conrad, Clifford L., Nancy J. Conrad and Harry B. Higley. Computer Mathematics. Rochelle Park, New Jersey: Hayden Book Company, Inc., 1975.

Golden, Neil. Computer Programming in the BASIC Language, second edition. New York: Harcourt, Brace, Jovanovich, 1981.
[outstanding text for 1 or 2 semester course]

Graham, Neill. Computers and Computing. St. Paul, Minnesota: West Publishing Co, 1982.
[good chapters on computer math and assembly and machine code]

Poirot, James L. and David N. Groves. Computers and Mathematics. Manchaca, Texas: Sterling Swift Publishing Company, 1979.
[good chapters on bases 2, 8, 16, computer circuits and logic]

SUPPLEMENTARY MATERIALS

Dale, Nell and David Orshalick. Introduction to PASCAL and Structured Design. Lexington, Maryland: D.C. Heath and Co., 1983.
[both of the above are good introductory PASCAL texts]

Findlay, William and David Watt. PASCAL, An Introduction to Methodical Programming. Rockville, Maryland: Computer Science Press, Inc., 1981.

Finkel, Leroy and Jerald R. Brown. Data File Programming in BASIC. New York: John Wiley and Sons, Inc., 1980.

Jacobs, French, Moulds, Schuchman. Computer Programming in the BASIC Language. Allyn and Bacon, 1978.
[good self-teaching text for teachers]

Presley, Bruce. A Guide to Programming in Applesoft. New York: Van Nostrand Reinhold Co., 1982.

Rogowski, Stephen J. Problems for Computer Solution. Morristown, New Jersey: Creative Computing Press, 1980.
[teachers's edition also available]

Shelly, Gary B. and Thomas J. Cashman. Introduction to BASIC Programing. Brea, California: Angheim Publishing Co., 1983
[good examples of flowcharting, good programming projects]

Spencer, Donald D. Problems for Computer Solution. Ormond Beach, Florida: Camelot Publishing Co., 1977.
[hundreds of problems]

EVALUATIVE TECHNIQUES IN COMPUTER SCIENCE

Evaluation is based on two key areas: assignments and tests. Each of these is in turn based on two types: factual information and programming. Thus we have a four-entry matrix that summarizes all the possibilities.

	Tests	Assignments
Factual		
Programming		

(1) (Factual, Tests): Tests of this nature are best suited to the non-programming parts of the course and to the early stages of programming. These techniques include short answer questions, fill in the blanks, and true/false. Units on the history of computers, uses of computers, and computer arithmetic are well suited to objective questions.

These techniques can also be employed early in the programming phase of a course by supplying a student with a list of computer statements with errors for him to correct. Also, supplying a correct program for the student to trace and write down the expected output fits this style of testing.

(2) (Programming, Tests): Tests that involve original programming put a student in a high-pressure situation and should be used with care. These tests are basically of three types:

(a) complete program -- the student is to write a completely documented program to solve a specific problem.

(b) program segments -- small pieces of code are written to demonstrate ability to solve very specific, short problems.

(c) alter a program -- the student is given a complete program with the instruction to make it do something slightly different from what it was originally designed to do.

(3) (Factual, Assignments): Textbooks are full of short, problem-types of questions to accompany every phase of programming. Drawing flowcharts, doing arithmetic in bases, and researching and reporting on history, uses, and types of computers are other sources of assignment material.

(4) (Programming, Assignments): This is certainly the heart of any programming-oriented computer course. Assignments can be given to individuals or pairs, but should be diverse enough so that almost everyone is working on a different problem. Grading of programs should be based on correctness, structure, and level of difficulty.